GCLC — A Tool for Constructive Euclidean Geometry and More than That

Predrag Janičić
Faculty of Mathematics
University of Belgrade

International Congress of Mathematical Software
Castro Urdiales, Spain, September 1–3, 2006.

Roadmap

- Purposes, History, Basic Principles, Features
- GCLC Language and Samples
- Theorem Prover and Deduction Control
- GeoThms System and XML support
- Related Systems, Conclusions and Future Work

Roadmap

- **Purposes, History, Basic Principles, Features**
- GCLC Language and Samples
- Theorem Prover and Deduction Control
- GeoThms System and XML support
- Related Systems, Conclusions and Future Work

The Main Purposes of GCLC/WinGCLC

- Dynamic geometry tool
- Visualizing geometry (and not only geometry)
- Producing digital mathematical illustrations of high quality
- Use in mathematical education, in studying geometry and as a research tool

Name of the Game

- Originally, a tool for producing geometrical illustrations for LaTeX, hence the name GCLC:

  "Geometry Constructions $\rightarrow$ LaTeX Converter".

GCLC: History and Releases

- Freely available releases for Windows, Linux

- Available from http://www.matf.bg.ac.yu/~janicic/gclc and from EMIS (The European Mathematical Information Service) servers http://www.emis.de/misc/index.html

- Hundreds of users worldwide

- First release in 1996, Windows GUI in 2003, theorem prover built-in in 2006

- Written in C/C++, around 20000 lines of code

GCLC: Basic Principles

- A construction is a formal procedure, not an image
- Producing mathematical illustrations should be based on "describing figures", not on "drawing figures" (similarly as $\TeX$)
- Images can be produced from descriptions, but not vice-versa!
- All instructions are given explicitly, in GCLC language
- GCLC language is like a simple programming language, easily understandable to mathematicians

Features (part I)

- Support for geometrical constructions: sequences of primitive construction steps performed by ruler and compass
- Support for compound constructions and transformations
- Symbolic expressions, while-loops, user-defined procedures
- Conics, 2D and 3D curves, 3D surfaces
- Built-in theorem prover

Features (part II)

- User-friendly interface, interactive work, animations, traces
- Import from JavaView
- Export to different formats (LaTeX, EPS, BMP, SVG)
- Full XML support
- Free, small in size (430Kb–830Kb), easy to use

Roadmap

- Purposes, History, Basic Principles, Features
- **GCLC Language and Samples**
- Theorem Prover and Deduction Control
- GeoThms System and XML support
- Related Systems, Conclusions and Future Work

GCLC Language (part I)

- Instructions for describing **content**
- Instructions for describing **presentation**
- All of them are explicit, given within GCLC documents

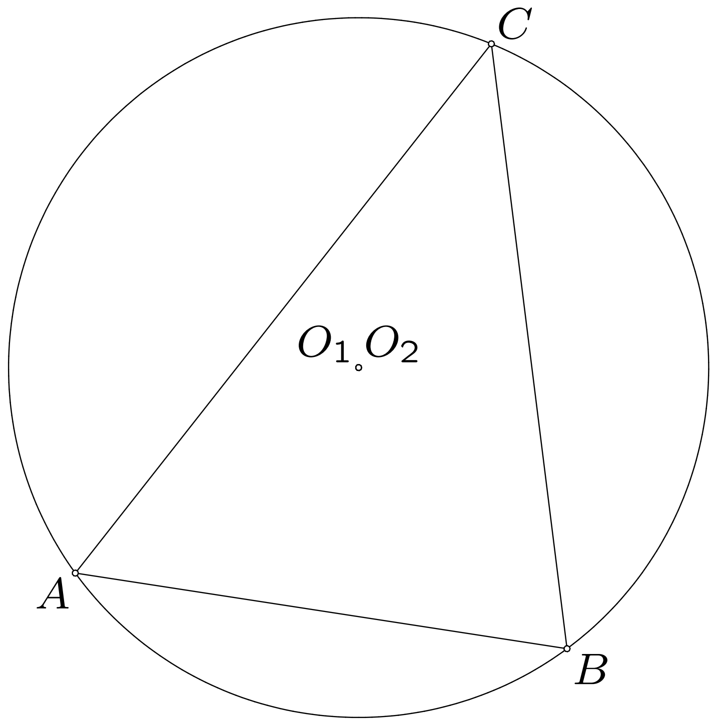
GCLC Language (part II)

- Basic definitions, constructions, transformations
- Drawing, labelling, and printing commands
- 2D and 3D Cartesian commands
- Symbolic expressions, loops, user-defined procedures
- Commands for describing animations
- Commands for the geometry theorem prover

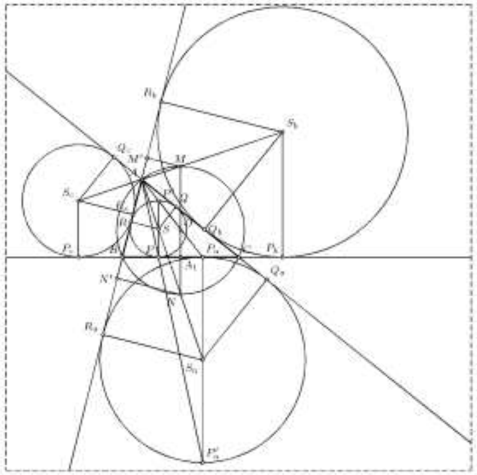
Simple Example (part I)

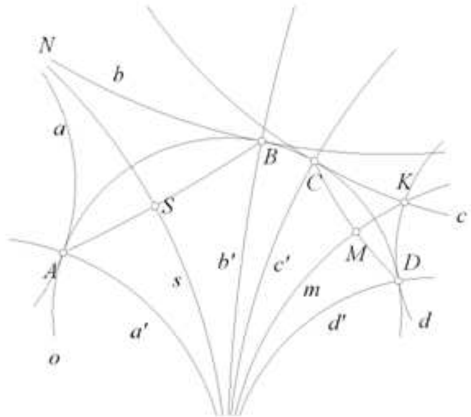
```
% fixed points                   |    % labelling points
point A 15 20                    |    cmark_lb A
point B 80 10                    |    cmark_rb B
point C 70 90                    |    cmark_rt C
                                 |    cmark_lt O_1
% side bisectors                 |    cmark_rt O_2
med a B C                        |
med b A C                        |    % drawing the sides of the triangle ABC
med c B A                        |    drawsegment A B
                                 |    drawsegment A C
% intersections of bisectors     |    drawsegment B C
intersection O_1 a b             |
intersection O_2 a c             |    % drawing the circumcircle of the triangle
                                 |    drawcircle O_1 A
```

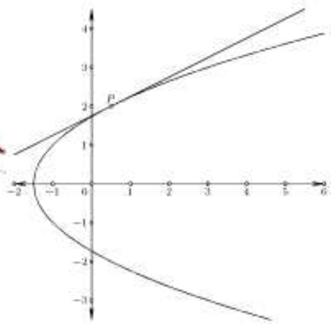
Simple Example (part II)

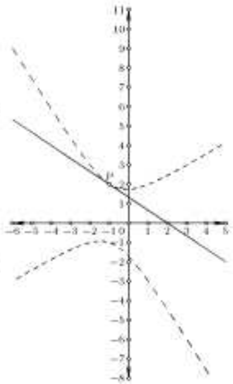


Samples

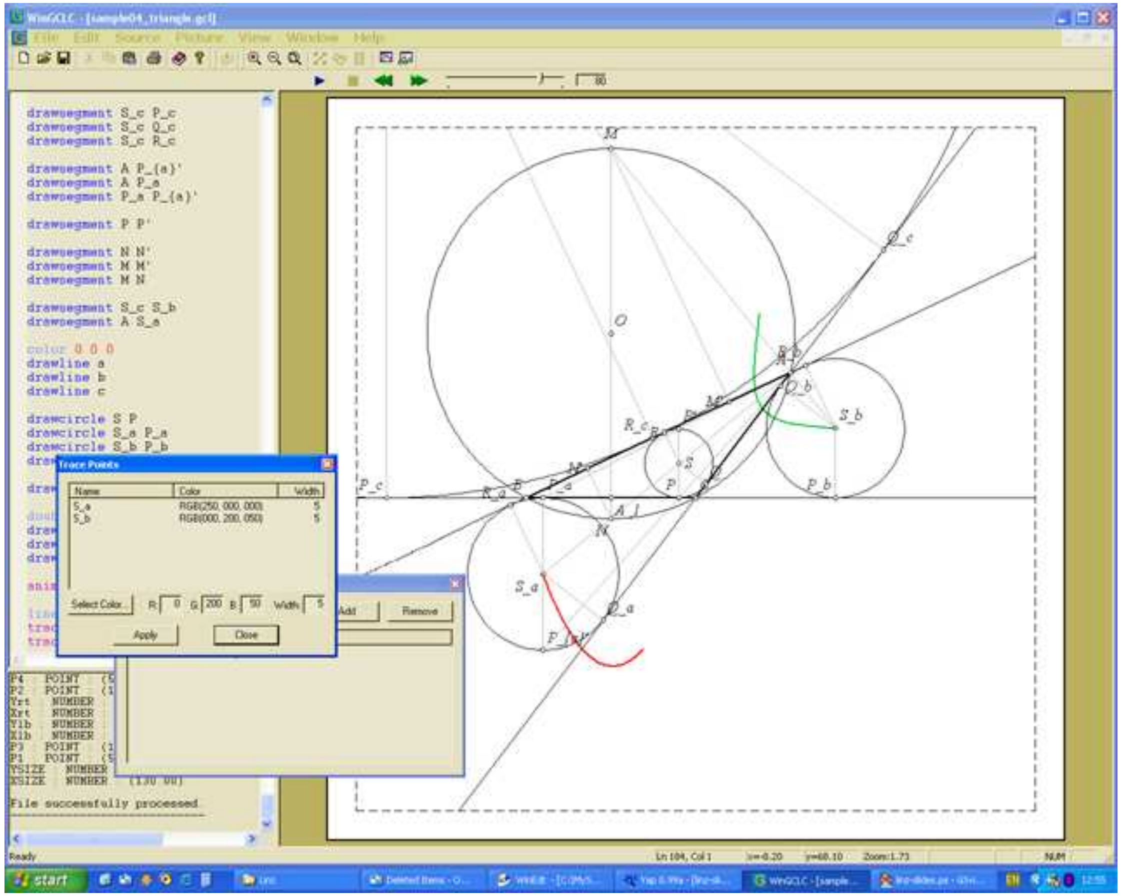








WinGCLC Screenshot



Roadmap

- Purposes, History, Basic Principles, Features
- GCLC Language and Samples
- **Theorem Prover and Deduction Control**
- GeoThms System and XML support
- Related Systems, Conclusions and Future Work

Built-in Theorem Prover

- Joint work with Pedro Quaresma, University of Coimbra
- Based on the area method (Chou et. al., mid 90's)
- Produces synthetic, coordinate-free, traditional, human-readable proofs
- Proofs generated in LaTeX with explanations for each step
- The prover tightly integrated into GCLC

Properties of the Area Method

- Wide realm, covers many non-trivial theorems

- Efficient for many non-trivial theorems

- Conjectures expressed in terms of equalities over geometry quantities — e.g., signed area of a triangle (S_{ABC}) and Pythagoras difference ($P_{ABC} = AB^2 + CB^2 - AC^2$)

- Current expression is transformed step by step, by different simplifications

All Proof Steps Are Explicit

- Elimination steps (elimination of constructed points in reverse order, by using appropriate lemmas)

- Algebraic simplifications (e.g., $x+0 \rightarrow x$, $\frac{x}{y}+\frac{u}{v} \rightarrow \frac{x \cdot v+u \cdot y}{y \cdot v}$)

- Geometric simplifications (e.g., $P_{AAB} \rightarrow 0$, $S_{ABC} \rightarrow S_{BCA}$)

- Proofs given in layers

Algebraic Simplifications

- Stand alone system

- Based on (around 40) rewrite rules (divided and applied in 20 groups)

- Simplification is sound and terminating. It leads to equalities of the form:

$$a_{1,1} \cdot a_{1,2} \cdot \ldots \cdot a_{1,n_1} + \ldots + a_{m,1} \cdot a_{m,2} \cdot \ldots \cdot a_{m,n_m} =$$
$$= \; b_{1,1} \cdot b_{1,2} \cdot \ldots \cdot b_{1,k_1} + \ldots + b_{l,1} \cdot b_{l,2} \cdot \ldots \cdot b_{l,k_l}$$

...and, finally, given $a_{i,j}$ and $b_{i,j}$ are independent values, the above form simplifies to *true* or *false*

Using the Theorem Prover

- For the given example, points `O_1` and `O_2` are identical. This can be stated as follows

```
prove { identical O_1 O_2 }
```

or

```
prove { equal
            { pythagoras_difference3 O_1 O_2 O_1 }
            { 0 }
      }
```

Fragment of the Proof

(113) $(0.062500 \cdot (P_{CBC} \cdot S_{BAC})) = \left(\frac{1}{4} \cdot \left(P_{CBM_a^0} \cdot S_{BAM_a^0}\right)\right)$, by algebraic simplifications

(114) $(0.062500 \cdot (P_{CBC} \cdot S_{BAC})) = \left(\frac{1}{4} \cdot \left(\left(P_{CBB} + \left(\frac{1}{2} \cdot (P_{CBC} + (-1 \cdot P_{CBB}))\right)\right) \cdot S_{BAM_a^0}\right)\right)$, by Lemma 29 (point M_a^0 eliminated)

(115) $(0.062500 \cdot (P_{CBC} \cdot S_{BAC})) = \left(\frac{1}{4} \cdot \left(\left(0 + \left(\frac{1}{2} \cdot (P_{CBC} + (-1 \cdot 0))\right)\right) \cdot S_{BAM_a^0}\right)\right)$, by geometric simplifications

(116) $(0.062500 \cdot S_{BAC}) = \left(\frac{1}{8} \cdot S_{BAM_a^0}\right)$, by algebraic simplifications

(117) $(0.062500 \cdot S_{BAC}) = \left(\frac{1}{8} \cdot \left(S_{BAB} + \left(\frac{1}{2} \cdot (S_{BAC} + (-1 \cdot S_{BAB}))\right)\right)\right)$, by Lemma 29 (point M_a^0 eliminated)

(118) $(0.062500 \cdot S_{BAC}) = \left(\frac{1}{8} \cdot \left(0 + \left(\frac{1}{2} \cdot (S_{BAC} + (-1 \cdot 0))\right)\right)\right)$, by geometric simplifications

(119) $0 = 0$, by algebraic simplifications

Experimental Results

Theorem Name	elim.steps	geom.steps	alg.steps	time
Ceva	3	6	23	0.001s
Gauss line	14	51	234	0.029s
Thales	6	18	34	0.001s
Menelaus	5	9	39	0.002s
Midpoint	8	19	45	0.002s
Pappus' Hexagon	24	65	269	0.040s
Ratio of Areas of Parallelograms	62	152	582	0.190s
Triangle Circumcircle	50	104	43	0.028s
Distance of a line containing the centroid to the vertices	274	673	3196	8.364s

One Application of the Theorem Prover: Automatic Verification of Regular Constructions

- The system for automated testing whether a construction is regular or illegal

- For instance — constructing a line l determined by (identical points) O_1 and O_2 from the above example is not a regular construction step

- Test is made by the theorem prover and the argument is given as a synthetic proof (the only such geometry tool?)

Invoking Automatic Verification of Regular Constructions

- Above example: constructing a line l determined by (identical points) O_1 and O_2 is not a regular construction step

- ```
  Error 14: Run-time error: Bad definition. Can not determine
  intersection. (Line: 26, position: 10) File not processed.

  Deduction check invoked: the property that led to the error
  will be tested for validity.

  Total number of proof steps:           18

  Time spent by the prover: 0.001 seconds
  The conjecture successfully proved - the critical property always holds.
  ```
  ```

Processing Descriptions of Constructions

- Syntactical check

- Semantical check (e.g., whether two concrete points determine a line)

- Deductive check, thanks to the verification mechanism (e.g., whether two constructed points never determine a line)

Roadmap

- Purposes, History, Basic Principles, Features
- GCLC Language and Samples
- Theorem Prover and Deduction Control
- **GeoThms System and XML support**
- Related Systems, Conclusions and Future Work

GeoThms

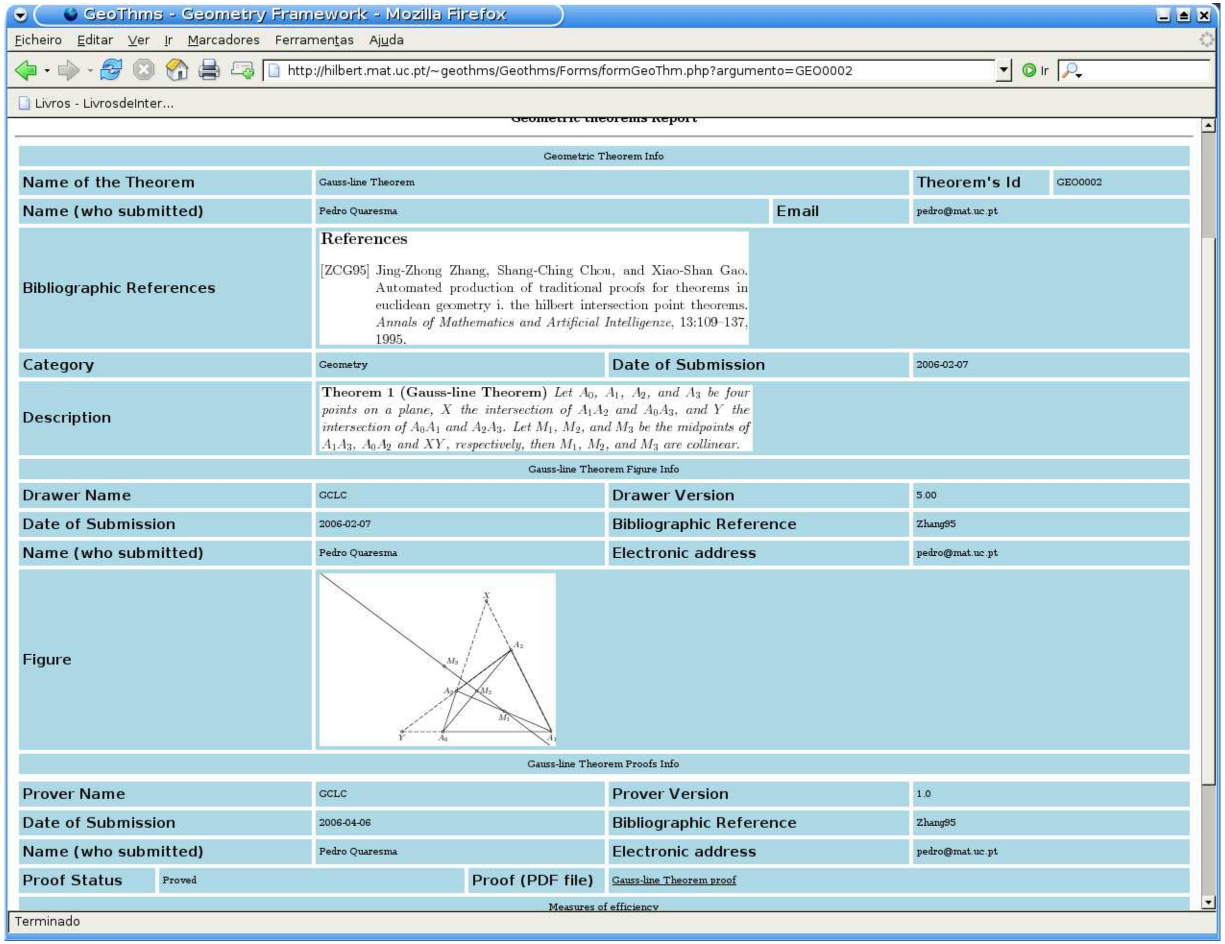
- Main author: Pedro Quaresma (University of Coimbra)

- An Internet framework that links dynamic geometry software (GCLC, Eukleides), geometry theorem provers (GCLCprover), and a repository of geometry problems (geoDB)

- A user can easily browse through the list of geometric problems, their statements, illustrations and proofs

- http://hilbert.mat.uc.pt/~geothms

GeoThms Screenshot



XML Support

- Format for descriptions of constructions and proofs

- Potentially common interchange format for different tools for geometrical constructions (one additional theorem prover by other authors already added to GeoThms)

- Web presentation, in different forms (e.g., natural language form).

Construction in XML form

Description of construction:

Let us define the following fixed points:

- Let *P1* be a point with Cartesian coordinates (5.000000, 5.000000).
- Let *P3* be a point with Cartesian coordinates (125.000000, 125.000000).

Let us draw the following objects:

- Visible area: left-bottom corner (5.000000,5.000000), right-top corner (125.000000, 125.000000).

Let us define the following fixed points:

- Let *P2* be a point with Cartesian coordinates (125.000000, 5.000000).
- Let *P4* be a point with Cartesian coordinates (5.000000, 125.000000).

Let us draw (using dashed style) the following objects:

- The segment with endpoints *P1* and *P2*.
- The segment with endpoints *P2* and *P3*.
- The segment with endpoints *P3* and *P4*.
- The segment with endpoints *P4* and *P1*.

Let us define the following fixed points:

- Let *B* be a point with Cartesian coordinates (35.000000, 60.000000).
- Let *C* be a point with Cartesian coordinates (65.000000, 60.000000).
- Let *A* be a point with Cartesian coordinates (40.000000, 80.000000).

Let us construct the following objects:

Proof in XML form

Step 1

(3.000000) * (s4(A, B, C, D)) = (13.000000) * (s3(A, B, A_2))

the statement

Semantic values: 4500.000000 = 4500.000865

Step 2

(3.000000) * (s4(A, B, C, D)) = (13.000000) * ((((s3(A, B, B_1)) * (s3(A, B, A_1))) + ((-1.000000) * ((s3(A_1, B, B_1)) * (s3(A, B, A)))))) / (s4(A, B, A_1, B_1)))

Lemma 30 (point A_2 eliminated)

Semantic values: 4500.000000 = 4500.000865

Step 3

(3.000000) * (s4(A, B, C, D)) = (13.000000) * ((((s3(A, B, B_1)) * (s3(A, B, A_1))) + ((-1.000000) * ((s3(A_1, B, B_1)) * (0.000000)))) / (s4(A, B, A_1, B_1)))

Lemma 2 (equal)

Semantic values: 4500.000000 = 4500.000865

Step 4

(3.000000) * (s4(A, B, C, D)) = (13.000000) * ((((s3(A, B, B_1)) * (s3(A, B, A_1))) + ((-1.000000) * (0.000000))) / (s4(A, B, A_1, B_1)))

multiplication by 0

Semantic values: 4500.000000 = 4500.000865

Step 5

(3.000000) * (s4(A, B, C, D)) = (13.000000) * ((((s3(A, B, B_1)) * (s3(A, B, A_1))) + (0.000000)) / (s4(A, B, A_1, B_1)))

multiplication by 0

Semantic values: 4500.000000 = 4500.000865

Roadmap

- Purposes, History, Basic Principles, Features
- GCLC Language and Samples
- Theorem Prover and Deduction Control
- GeoThms System and XML support
- **Related Systems, Conclusions and Future Work**

Related Systems

- Dynamic geometry tools: Cinderella, Geometer's Sketchpad, Eukleides, Cabri, JavaView, KSEG, . . .

- Tools with geometry theorem provers: GEX, GeoTher, Geometry Explorer, Theorema, Coq, MMP Geometer

- Repositories of geometrical problems: geometriagon

As a Conclusion: Main Applications

- Producing digital mathematical illustrations
- Storing mathematical contents
- Mathematical education
- GeoThms: a major Internet resource for geometrical problems

Further Work

- Support for additional mathematical objects (e.g., graphs and flowcharts), leading to a general-purpose mathematical illustration tool

- Additional automated reasoners (not only geometrical)

- Enabling moving along/packing/unpacking parts of proofs

- Additional options for interactive work, especially developing automatic tutors

- Import from printed and hand-made constructions